Scheduling Optimization with LDA and Greedy Algorithm

by

Yongjian Bi

Department of Statistical Science
Duke University

Date: _____

Approved:

_____
David L. Banks, Supervisor

_____
Sayan Mukherjee, Chair

_____
Surya Tokdar

Thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science in the Department of Statistical Science
in the Graduate School of Duke University
2016

Abstract

Scheduling Optimization with LDA and Greedy Algorithm

by

Yongjian Bi

Department of Statistical Science
Duke University

Date: _____

Approved:

_____
David L. Banks, Supervisor

_____
Sayan Mukherjee, Chair

_____
Surya Tokdar

An abstract of a thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in the Department of Statistical Science in the Graduate School of Duke University
2016

Abstract

Scheduling optimization is concerned with the optimal allocation of events to time slots. In this paper, we look at one particular example of scheduling problems - the 2015 Joint Statistical Meetings. We want to assign each session among similar topics to time slots to reduce scheduling conflicts. Chapter 1 briefly talks about the motivation for this example as well as the constraints and the optimality criterion. Chapter 2 proposes use of Latent Dirichlet Allocation (LDA) to identify the topic proportions in each session and talks about the fitting of the model. Chapter 3 translates these ideas into a mathematical formulation and introduces a Greedy Algorithm to minimize conflicts. Chapter 4 demonstrates the improvement of the scheduling with this method.

Contents

List of Tables

List of Figures

Acknowledgements

I want to thank my advisor, David Banks, for his kind support and encouragement. The thesis would not be possible without his help. And I also would like to thank Kamesh Munagala for providing valuable insights and comments.

1

Introduction

The Joint Statistical Meetings are one of the largest statistical events in the world, with more than 6000 attendees, 600 sessions and 75 employers hiring for more than 200 positions. Each session is planned in advance by one or more organizers and includes speakers presenting on a common subject. Each speaker has about 20 minutes to present, but some special events allow more time. The subjects of the sessions range from statistical applications to methodology and theory to the expanding boundaries of statistics, such as analytics and data science. However, with so many sessions and attendees, a person cannot go to all of the sessions that interest him/her because, within a given time slot, some sessions with similar topics take place at the same time. For example, if two sessions on clinical trials are scheduled at the same time, a person can only go to one of them.

To quantify the conflicts between each pair of sessions, we need to understand the topic mixture of the sessions. In chapter 2, we propose using Latent Dirichlet Allocation (LDA) to identify the topic proportions of each session and describe some key steps in standardizing the text data. After that, we demonstrate the performance of the topic modeling by showing both the token distributions for some topics and

the topic distributions for some sessions.

Then, in chapter 3, we talk about the optimization problem again, detailing the criteria and constraints. Then we translate these ideas into a mathematical formulation and introduce a brute force method for minimizing conflicts. This result quantifies the improvement over the original schedule.

2

Topic Modeling of JSM sessions with LDA

2.1 LDA

The Latent Dirichlet Allocation (LDA) model described in Blei et al. [6] is a statistical method that tries to identify topics in texts. Many people have been using it as a tool for natural language processing.

LDA makes a couple of assumptions. First, we have a fixed finite vocabulary of terms $W = \{w_1, ..., w_v\}$. Also, each document is a set of terms $D = d_1, ..., d_n$ where $d_i \in W$, and the corpus is defined as the set of documents $D_1, ..., D_m$ being analyzed. We can use LDA to represent the documents as mixtures of latent topics, which are fitted as unknown multinomial distributions over the vocabulary that needs to be inferred from the data. The most fundamental LDA assumption is that the number of unknown latent topics is a fixed finite integer K. The only data are the word counts from the documents.

The data generating process for the LDA model can be described as follows [1]. And this generative process can be visually represented with the plate diagram Figure 2.1.

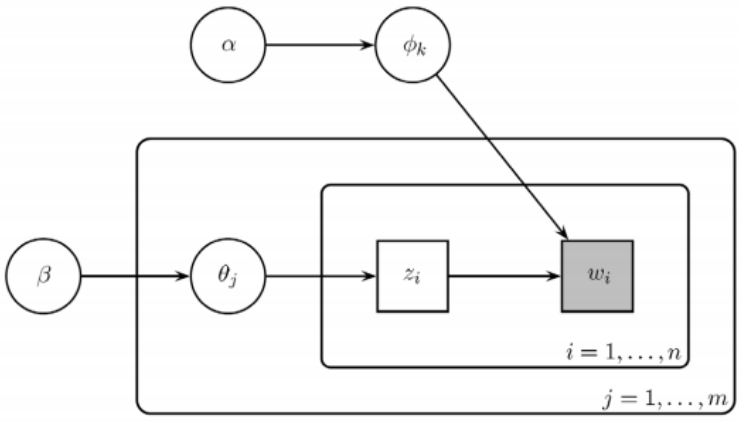

FIGURE 2.1: The plate diagram for the basic LDA generative process (figure from Soriano et al. [1]

1. For each topic k:

   - Draw a vector of word proportions $\phi_k \sim Dir_V(\alpha)$. This determines the relative weights of each term in topic k.

2. For each document D_j:

   - Draw a vector of topic proportions $\theta_j \sim Dir_j(\beta)$. This determines the extent to which document D_j is composed of each of the K topics.
   - For each word w_i in document j: Draw a topic assignment $z_i \sim Mult(\theta_j)$. Draw a word from the topic $w_i \sim Mult(\phi_{z_i})$.

The goal of LDA is to invert the generative model and find the posterior distribution of the latent variables conditional on the documents.

$$p(\theta_{1:m}, z_{1:m,1:n}, \phi_{1:K}|w_{1:m,1:n}, \alpha, \beta) = \frac{p(\theta_{1:m}, z_{1:m}, \phi_{1:K}|w_{1:m}, \alpha, \beta)}{\int_{\phi_{1:K}} \int_{\theta_{1:m}} \sum_{z_{1:m,1:n}} p(\theta_{1:m}, z_{1:m}, \phi_{1:K}|w_{1:m}, \alpha, \beta)} \tag{2.1}$$

In (2.1), the denominator is intractable, so the solution requires Markov Chain Monte Carlo or variational inference methods. For example, Blei et al. [6] used a

variational Bayes approximation while Griffiths and Steyvers [5] proposed a collapsed Gibbs sampler that integrates out the θ_j and ϕ_k.

Besides this topic model, researchers have also proposed numerous extensions of topic modeling. For example, Blei et al. [7] proposed hierarchical LDA to connect topics together in a hierarchy related to the nested Chinese restaurant process. Also, Blei and Lafferty [8] proposed a dynamic extension to topic models to analyze the time evolution of topics. Meanwhile, Blei and McAuliffe [9] introduced supervised LDA to jointly model the topics of a corpus and a response variable together that is associated with each document. Moreover, Chang and Blei [10] proposed the relational topic model to connect a network of documents, find links, and predict words within documents.

For the session abstracts, I fit only the basic LDA model with 20 latent topics because the top topic words are more easy to interpret. Alternatively, one can try out a different number of topics, select the one which has the largest likelihood. Or instead of using LDA, one may use Hierarchical Dirichlet Process (HDP) LDA if the corpus size is relatively small, which is designed to address the case where the number of mixture components (the number of topics) is not known priori [11]. If the size of the corpus is too big to fit HDP-LDA, one can draw a smaller uniform sample of the corpus and use the number of topics given by HDP-LDA on that subsample. Several other methods are also available [2; 3; 4]

2.2 Session Abstracts Data

The data are downloaded from the JSM website. Each abstract is associated with a unique abstract key number, a non-unique session number, author names, title, and time of that session. In 2015, JSM had more than 3000 abstracts and 700 sessions. We scraped the data with Python's built-in method of request and HTML. We created a function in Python to remove unnecessary characters, such as backslash

n, backslash r and backslash t. Besides, we have written regular expression functions to extract the relevant field of values. We also set the time interval of each request to be 1 second to avoid causing any burden on the server of the JSM website.

2.3 Cleaning Data

The cleaning and standardization of the data consists of the following steps:

1. Remove all stop words.
2. Stem the words so that sing/sang/sung/song are all mapped to the same token.
3. Remove tokens that only appear once.
4. Remove tokens that only appear in the abstracts for one session.
5. Create n-grams, so that *time series* or *maximum likelihood estimate* are mapped to one token.
6. Calculate the TF-IDF score for each token-document combination, and then find the variance in TF-IDF scores for each token. Rank from high to low variance and use expert judgment to discard low variance terms.
7. Discard tokens that do not appear more than ten times.
8. Do 1, 2, 5, 6 on two books called *Little Busybodies* by Jeanette Augustus Marks and Julia Moodyand and *Herein is Love* by Reuel L. Howe. Remove all the tokens in session abstract data that appear in these two books.

   For step 1, David Blei has a good list of stop words; it is part of his Python implementation of turbotopics on his website. For step 2, we used the extension called SnowballStemmer from natural language processing toolkit in Python. Steps 3 and 4 are straightforward. Concerning step 5, we mostly focused on

finding bigrams and trigrams. We iterate through the entire corpus and add each pair of bigrams to the list; then we delete all of these bigrams that do not appear at least five times in the entire corpus. This process is similar to finding trigrams. After taking into account these bigrams, the single word token is also adjusted. For example, suppose that the word 'time' occurs 80 times after we include the bigram time-series, we should re-evaluate the token 'time' in the corpus and remove it if it is in the bigram time-series.

For step 6, TF-IDF is calculated as following: for a term i in document j:

$$w_{i,j} = tf_{i,j} * log(\frac{N}{df_i}) \tag{2.2}$$

where $tf_{i,j}$ = number of occurrences of i in j, df_i = number of documents containing i and N = total number of documents. After calculating the TF-IDF score for each token across all documents, we calculate the variance and delete tokens with very low variance. After the first seven steps, we found that there are still some n-grams that are not helpful in identifying the topics in the corpus, and we proposed using some books that are almost completely irrelevant to statistics to remove these unhelpful n-grams. They were downloaded from Project Gutenberg. The first one is called *Little Busybodies* by Jeanette Augustus Marks and Julia Moodyand. It is about learning the life of crickets, bees, beetles and other insects for pre-school children. The second one is called *Herein is Love* by Reuel L. Howe. It is about a study of the biblical doctrine of love in its bearing on personality, parenthood, teaching, and all other human relationships. After this step, we removed about 200 bigrams and 400 trigrams.

2.4 Topic Modeling Results

After the data cleaning step, we apply the LDA on our session abstract dataset. The top 7 tokens in the first six topics found by LDA are included in Table 2.1.

Table 2.1: The top 7 words in the first 6 topics found by LDA for the session abstracts dataset

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6
spatial	patient	prior	student	gene	test
region	health	factor	statist	express	method
locat	risk	bayesian	class	genom	power
climat	provid	posterior	cour	genet	sampl
predict	studi	infer	onlin	sequenc	size
process	care	comput	research	trait	type
uncertainti	outcom	estim	teach	variant	method

Interestingly, topic 1 appears to focus on the spatial statistics, which has keywords like location, region and climate. Topic 2 seems to be about providing better care and outcomes for patients using statistics. Topic 3 seems to about Bayesian statistics with keywords like posterior, prior and Bayesian. Topic 4 seems to statistical pedagogy with keywords like student, class, course and teaching. Topic 5 seems to be about statistics for genomic data with keywords like gene, express and sequence. Topic 6 looks a lot like hypothesis testing with keywords like power, test, size and type.

Additionally, from figure 2.2, we can see that most sessions have a mix of no more than two major topics.

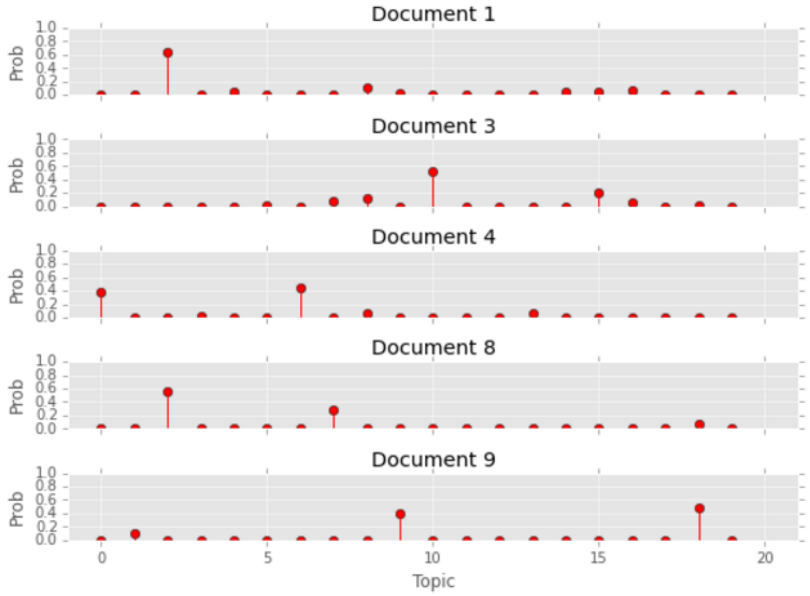

FIGURE 2.2: Topic distribution for selected sessions

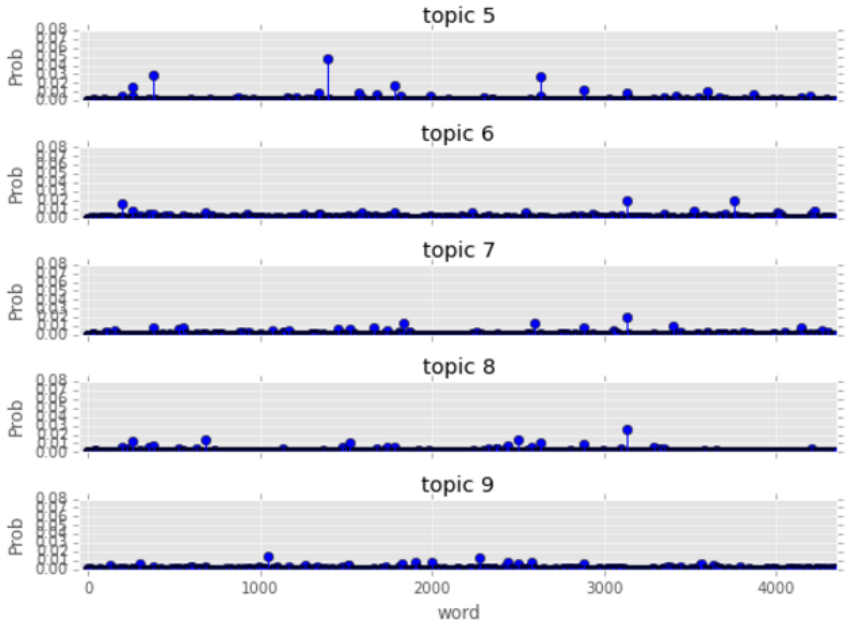

FIGURE 2.3: Token distribution for selected topics

3

Mathematical Formulation

After we have discovered the topic proportions in each session, we come back to the optimization problem again. The constraint in this problem is that each session has to take place exactly once, and there cannot be more than $N = 46$ sessions at a given time.

We can translate the context of the problem and constraints into the following mathematical formulation:

$$\begin{aligned}
\underset{X}{\text{Minimize}} \quad & \sum_{i \neq j} X_i \cdot X_j * I(S_i, S_j) \\
\text{subject to} \quad & \forall i, \sum_j X_i \cdot X_j \leqslant N \\
& \forall i, X_i \cdot X_i = 1 \\
& I(S_i, S_j) = f(\max_{\sigma} \min(p_{i\sigma}, p_{j\sigma})) \\
& f(x) = \begin{cases} 3x, & \text{if } 0.5 < x \\ x, & \text{otherwise} \end{cases}
\end{aligned}$$

where $I(S_i, S_j)$ is defined as an interference function between two sessions and σ indexes the topic. Suppose we have three topics, and the topic vector for one session is

$[0, 6, 0, 3, 0.1]$, another vector might be $[0, 3, 0, 6, 0.1]$. Then the interference between these two sessions, if they are within the same time slot, is $f(0.3)$. Similarly, for two sessions with topic vectors as $[0, 7, 0, 2, 0.1]$ and $[0, 6, 0, 3, 0.1]$, the interference is $f(0.6)$. One can change the function f herself according to the context of the optimization problem. In this case, I would like to have three times greater penalty if the conflict is bigger than 0.5 to avoid further having sessions with same dominant topics conducted at the same time. Each X_i is a m dimensional vector where m indicates the number of time slots. For example, $X_i = (0, 1, 0, 0, 0, 0)$ means that session i takes place during the second time slot of six time slots. if $X_i \neq X_j$ where $i \neq j$, then $X_i \cdot X_j = 0$.

3.1 Greedy Algorithm

A greedy algorithm is an algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage with the hope of finding a global optimum. In general, a greedy strategy does not necessarily lead to an optimal solution, but nonetheless, a greedy heuristic may give a locally optimal solution that approximates a global optimal solution in a reasonable amount of time. One way to circumvent the local optimum and have the global optimum is to have several random starting points to see whether they all lead to the same optimum. I will first outline the general steps of this approach and then some ways to avoid being stuck at the local minimum.

For iteration l from 1 through L:

1. Assign each session randomly to a time slot with the constraints mentioned at the beginning of this chapter, calculate $TF_l = \sum_{i \neq j} X_i \cdot X_j * I(S_i, S_j)$

2. For each session i:

(a) If session i interchanges time slots with session $j \neq i$, calculate TF_{ij}

(b) Among all TF_{ij}, find the j^* that has the minimum TF_{ij}, make session i interchange time slots with session j^*.

(c) If $TF_l > TF_{ij*}$, assign TF_{ij*} to TF_l, do (a),(b) and (c) with $i+1$. if not, stop the iteration, store TF_l and X_{il} for all session i

For $2(b)$, there might be multiple minima, so we propose randomly selecting one from all j^*. Moreover, for 2 (c), TF_l can only be bigger or equal to TF_{ij*}. So If TF_l is not greater than TF_{ij*}, we know we have achieved a local minimum and store the state of X and TF at that minimum. Once we have all k TF, we find the ones that have the minimum TF and see whether they lead to similar X. By 'similar', we mean that given a specific permutation of time slots, we end up with the same X. With multiple TF after the algorithm, we will have a sense of whether it is a good approximation of the global optimum and whether it is prone to be stuck in the local minimum. Moreover, one can do the embarrassingly parallel code to do L jobs with each leading to a local minimum.

3.2 Result

After we are done with the data cleaning and standardizing, we have 598 sessions and 13 time slots, with each slot having 46 sessions. We excluded the sessions that had 50 minutes duration but kept the sessions that are two hours long (598/680). Table 3.1 shows the number of sessions for each time slot originally.

After finishing the optimization, we decrease the total interference from about 1600 to 1200, which may not be the global minimum. However, the improvement is still very significant and we can use the optimized schedule for all two-hour sessions

From figure 3.1 and 3.2, we can see that time slots of first 13 sessions are changed

Table 3.1: Number of sessions for each time slot

1	2	3	4	5	6	7	8	9	10	11	12	13
44	50	42	39	44	47	63	58	43	47	42	53	26

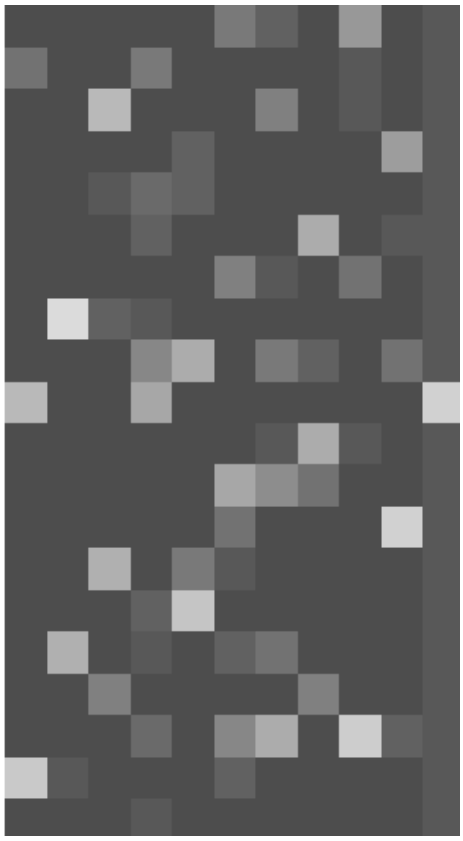

FIGURE 3.1: First 13 sessions topic distributions from the first time slot before optimization

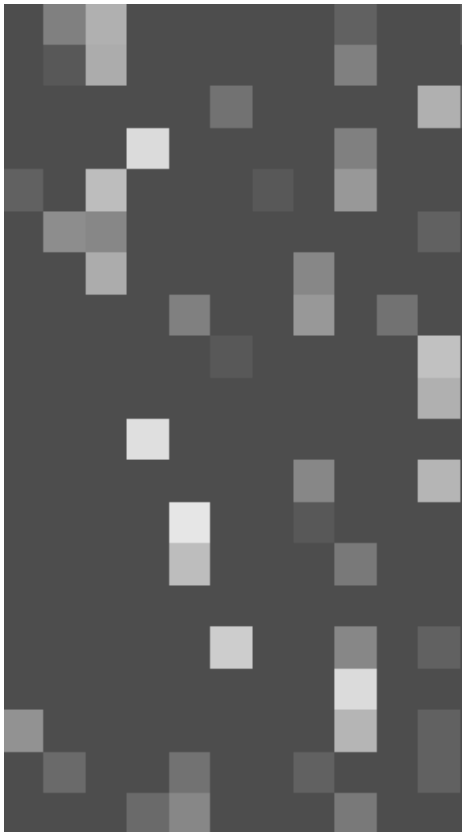

FIGURE 3.2: First 13 sessions topic distributions from the first time slot after optimization

to minimize interference. The original 13 sessions of the first time slot are no longer the same as the new ones. However, visualization is only a tool to help us confirm our result. We cannot solely base our judgment of whether scheduling is improved on visualization. That is why we introduce the total interference score to quantify the improvement.

4

Final Remarks

In summary, the extent of the improvement in the scheduling is mainly based on two strategies. One strategy is to identify the topic proportion of each session accurately. Identification of topic proportion not only depends on cleaning and standardizing the data but also on correctly identifying the number of latent topics. In this paper, the number of topics is discovered on ad-hoc basis - looking at the topic top words after fitting the model. Another problem that drives the extent of the improvement is to approximate the global minimum of optimization. This is done by starting from several random starting points that assign sessions to time slots. Besides, one can tune the function of the interference between two sessions according to the context of the optimization criteria.

Bibliography

[1] Soriano, Jacopo, Timothy Au, and David Banks. *Text mining in computational advertising.* 2013: Statistical Analysis and Data Mining 6.4 (2013): 273-285.

[2] Rajkumar Arun, V. Suresh, C. E. Veni Madhavan, and M. N. Narasimha Murthy *On Finding the Natural Number of Topics with Latent Dirichlet Allocation: Some Observations* 2010.

[3] Cao Juan, Xia Tian, Li Jintao, Zhang Yongdong, and Tang Sheng *A density-based method for adaptive LDA model selection* 2009: Neurocomputing 16th European Symposium on Artificial Neural Networks 2008 72, 79: 17751781.

[4] Romain Deveaud, Eric SanJuan, and Patrice Bellot *Accurate and effective latent concept modeling for ad hoc information retrieval.* 2014: Document numrique 17, 1: 6184.

[5] Thomas L. Griffiths and Mark Steyvers *Finding scientific topics.* 2004: Proceedings of the National Academy of Sciences 101, suppl 1: 52285235

[6] Blei, D. M., Ng, A. Y., and Jordan, M. I *Latent dirichlet allocation* 2003: Journal of Machine Learning Research, 3, 993-1022.

[7] Griffiths, D. M. B. T. L., and M. I. J. J. B. Tenenbaum. *Hierarchical topic models and the nested chinese restaurant process* 2004: JAdvances in neural information processing systems 16 (2004): 17.

[8] Blei, David M., and John D. Lafferty. *Dynamic topic models* 2006: Proceedings of the 23rd international conference on Machine learning. ACM, 2006.

[9] Mcauliffe, Jon D., and David M. Blei. *Supervised topic models* 2008: Advances in neural information processing systems

[10] Chang, Jonathan, and David M. Blei. *Hierarchical relational models for document networks* 2010: The Annals of Applied Statistics (2010): 124-150.

[11] Teh, Yee Whye, et al. *Hierarchical dirichlet processes* 2012: Journal of the American Statistical Association